Министерство образования и науки Российской Федерации

Федеральное агентство по образованию

Государственное образовательное учреждение

высшего профессионального образования

«Алтайский государственный технический университет им. И.И. Ползунова»

Кафедра *"* *Организация и безопасность движения "\_*

|  |  |
| --- | --- |
|  | Курсовая работа защищена с оценкой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Руководитель  работы *к.т.н., доцент Д.Ю. Каширский*  должность, и.о. фамилия |

Описать массивы. Написать программу, которая проверяет, является ли введенная с клавиатуры квадратная матрица «магическим» квадратом

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОЙ РАБОТЕ**

по дисциплине «Информатика»

*КР 190702.16.000ПЗ*

обозначение документа

Работу выполнил \_\_\_\_\_\_\_\_\_*М.Ю. Ляпин*

Студент гр. ОБД-81 подпись, и.о. фамилия

Барнаул - 2009

Изм.

№ докум.

Подпись

Дата

Лист

3

*КР 190702. 16. 000ПЗ*

Розраб.

Ляпин М.Ю.

Провер.

Каширский Д.Ю.

Утвер.

«Магический» квадрат

Лит.

Листов

25

АлтГТУ,ОБД-81

Содержание

Введение………………………………………………………………………………….4

1 Теоретическая часть…………………………………………………………………...6

1.1 История……………………………………………………………………………….6

1.2 Краткое описание Borland Delphi…………………………………………………...7

1.2.1 Описание окна Delphi 7…………………………………………………………...7

1.2.2 Описание окна Form1……………………………………………………………..7

1.2.2.1 Свойства объекта TForm………………………………………………………...8

1.2.3 Описание окна Object Inspector…………………………………………………..9

1.2.4 Описание окна Object TreeView…………………………………………………10

1.2.5 Описание окна Unitl.pas………………………………………………………….10

1.3 Обзор компонентов Borland Delphi………………………………………………..11

1.3.1 Описание компонента TEdit……………………………………………………..11

1.3.1.1 Свойства объекта TEdit………………………………………………………...11

1.3.2 Описание компонента TLabel……………………………………………………12

1.3.2.1 Свойства объекта TLabe………………………………………………………..12

1.3.3 Описание компонента TButton…………………………………………………..13

1.3.3.1 Свойства объекта TButton……………………………………………………...13

2 Программирование программы……………………………………………………...14

2.1 Программа…………………………………………………………………………..14

2.2 Постановка задачи………………………………………………………………….14

2.3 Этапы разработки программы……………………………………………………..15

2.3.1 Спецификация…………………………………………………………………….15

2.3.2 Разработка алгоритма…………………………………………………………….16

2.3.3 Кодирование………………………………………………………………………18

2.3.4 Массив……………………………………………………………………………..20

2.4 Анализ и тестирование……………………………………………………………..21

2.5 Создание справочной системы…………………………………………………….22

2.6 Создание установочного диска…………………………………………………….23

Заключение……………………………………………………………………………...24

Список литературы……………………………………………………………………..25

# **Введение**

Изм.

№ докум.

Подпись

Дата

Лист

4

*КР 190702. 16. 000ПЗ*

Предмет «Информатика» изучает компьютеры и информационные технологии. В свете быстро развивающегося технического процесса и все большего внедрения компьютеров в нашу жизнь этот предмет наряду с основными предметами, такими как математика и физика, приобретает все большее значение. Знание компьютеров и программирования уже необходимо не только на специфических предприятиях, но и на заводах и в простом офисе, где вся бухгалтерия ведется на компьютерах.

Информатика-это техническая наука, систематизирующая приемы создания, хранения, обработки и передачи информации средствами вычислительной техники, а также принципы функционирования этих средств и методы управления ими.

Из этого определения видно, что информатика очень близка к технологии, поскольку отвечает на вопрос как…?

* Как принимать и хранить информацию?
* Как обрабатывать информацию и преобразовывать ее в форму, удобную для человека?
* Как использовать вычислительную технику с наибольшей эффективностью?
* Как использовать достижения других наук для создания новых средств вычислительной техники?
* Как управлять техническими средствами с помощью программ?

Поэтому не случайно предмет, изучением которого занимается информатика, нередко называют информационной технологией или компьютерной технологией.

Важным в нашем определении является то, что у информатики есть как бы две стороны. С одной стороны, она занимается изучением устройств и принципов действия средств вычислительной техники, а с другой стороны - систематизацией приемов и методов работы с программами, управляющими этой техникой.

Итак, информатика-это наука, изучающая все аспекты получения, хранения, преобразования, передачи и использования информации.

Теоретическая информатика-это часть информатики, связанная с разработкой теорий, таких как: теория информации, теория алгоритмов, теория кодирования, теория автоматов, теория сетей и т.д., в том числе теоретических вопросов, связанных с проблемой создания искусственного интеллекта.

Целью данной курсовой работы является создание программного продукта. Эту программу я смог составить после того, как просмотрел несколько различных примеров создания подобных программ и изучил их основные принципы. Программа предназначена для проверки квадратной матрицы, введенной с клавиатуры, на «магический» квадрат, в зависимости от того какие числа будут введены. «Магическим» квадратом называется матрица, у которой сумма чисел в каждом горизонтальном ряду, в каждом вертикальном и по каждой из диагоналей одна и та же.

Лист

5

# **Теоретическая часть**

## **1.1 История**

Язык программирования – это способ записи программ решения различных задач на ЭВМ в понятной для компьютера форме.

Значительным событием в истории языков программирования стало создание в 1971 г. языка Паскаль. Его автор – швейцарский профессор

Н.Вирт – разрабатывал Паскаль как учебный зык структурного программирования.

Наибольший успех в распространении этого языка обеспечили персональные компьютеры. Фирма *Borland*

*International Inc* (США) разработала систему программирования Турбо Паскаль для ПК. Турбо Паскаль – это не только язык и транслятор с него, но еще и операционная оболочка, обеспечивающая пользователю удобство работы. Турбо Паскаль вышел за рамки учебного предназначения и стал языком профессионального программирования с универсальными возможностями.

Delphi 2007 — это современная система программирования, в основе которой лежит объектно-ориентированный язык Раsса1 и визуальные средства программирования. Система Delphi 2007 относится к классу средств ускоренной разработки программ. Это ускорение достигается за счет двух характерных свойств Delphi: визуального конструирования форм и использования библиотеки визуальных компонентов. Еще одно достоинство Delphi — мощность и гибкость. Это неудивительно, так как в основе Delphi лежит мощный объектно-ориентированный язык Object Раsса1, пригодный для создания программ любой сложности. Кроме того, программы, созданные в среде Delphi 2007, могут работать не только под управлением Windows.

Лист

6

## **1.2 Краткое описание Borland Delphi**

## Delphi 7 состоит из 5 окон:

* главное окно — **Delphi 7;**
* окно стартовой формы — **Form 1;**
* окно редактора свойств объектов — **Object Inspector;**
* окно просмотра списка объектов — **Object TreeView;**
* окно редактора кода — **Unitl.pas.**

**1.2.1 Описание окна Delphi 7**

В главном окне (рисунок 1) находится меню команд, панели инструментов и палитра компонентов.

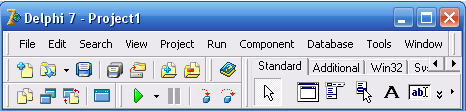


Рисунок 1 – главное окно

**1.2.2 Описание окна Form 1**

Окно стартовой формы **-Form1** (рисунок 2) представляет собой заготовку главного окна разрабатываемого *приложения.*



Рисунок 2 – окно Form1

Лист

7

**1.2.2.1Свойства объекта TForm**

Работа над *новым проектом,* так в Delphi называется разрабатываемое приложение, начинается с создания стартовой формы. Так на этапе разработки программы называют диалоговые окна.

Стартовая форма создается путем изменения значений свойств формы **Form1** и добавления к форме необходимых компонентов (полей ввода и вывода текста, командных кнопок).

Свойства формы, приведенные в таблице 1, определяют ее внешний вид: размер, положение на экране, текст заголовка, вид рамки.

Для просмотра и изменения значений свойств формы и ее компонентов используется окно **Object Inspector.** В верхней части окна **Object Inspector** указано имя объекта, значения свойств которого отображается в данный момент. В левой колонке вкладки **Properties** (Свойства) перечислены свойства объекта, а в правой — указаны их значения.

Таблица 1 – Свойства объекта TForm

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| Name | Имя формы. В программе имя формы используется для управления формой и доступа к компонентам формы |
| Caption | Текст заголовка |
| Width | Ширина формы |
| Height | Высота формы |
| Left | Расстояние от левой границы формы до левой границы экрана |
| BorderStyle | Вид границы. Граница может быть обычной (bsSizeable), тонкой (bsSingle) или отсутствовать (bsNone). Если у окна обычная граница, то во время работы программы пользователь может при помощи мыши изменить размер окна. Изменить размер окна с тонкой границей нельзя |
| Icon | Значок в заголовке диалогового окна, обозначающий кнопку вывода системного меню |
| Color | Цвет фона. Цвет можно задать, указав название цвета или привязку к текущей цветовой схеме операционной системы. Во втором случае цвет определяется текущей цветовой схемой, выбранным компонентом привязки и меняется при изменении цветовой схемы операционной системы |

8

Лист

9

Лист

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| Font | Шрифт. Шрифт, используемый "по умолчанию" компонентами, находящимися на поверхности формы. Изменение свойства Font формы приводит к автоматическому изменению свойства Font компонента, располагающегося на поверхности формы. То есть компоненты наследуют свойство Font от формы (имеется возможность запретить наследование) |
| Borderlcons | Кнопки управления окном. Значение свойства определяет, какие кнопки управления окном будут доступны пользователю во время работы программы. Значение свойства задается путем присвоения значений уточняющим свойствам biSystemMenu, biMinimaze, biMaximaze и biHelp. Свойство biSystemMenu определяет доступность кнопки. Свернуть и кнопки системного меню, biMinimaze— кнопки. Свернуть, biMaximaze— кнопки Развернуть, biHelp — кнопки вывода справочной информации |

**1.2.3 Описание окна Object Inspector**

Окно **Object Inspector** (рисунок 3) — окно редактора свойств объектов предназначено для редактирования значений свойств объектов. В терминологии визуального проектирования *объекты —* это диалоговые окна и элементы управления (поля ввода и вывода, командные кнопки, переключатели и др.). *Свойства объекта —* это характеристики, определяющие вид, положение и поведение объекта. Например, свойства Width и Height задают размер (ширину и высоту) формы, свойства тор и Left — положение формы на экране, свойство Caption — текст заголовка.

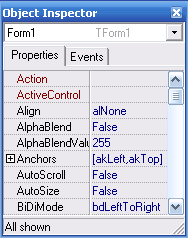


Рисунок 3 - Окно Object Inspector

10

Лист

**1.2.4 Описание окна Object TreeView**

В окне просмотра списка объектов- **Object TreeView**(рисунок 4), указаны все объекты, использовавшиеся при создании программы.

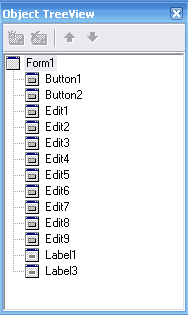


Рисунок 4 - Окно Object TreeView

**1.2.5 Описание окна Unitl.pas**

В окне редактора кода (рисунок 5), которое можно увидеть, отодвинув в сторону окно формы, следует набирать текст программы. В начале работы над новым проектом это окно редактора кода содержит сформированный Delphi шаблон программы.

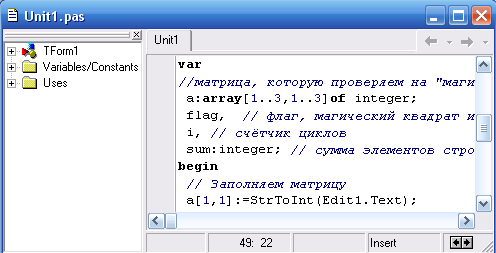


Рисунок 5 - Окно Unitl.pas

1.3 [Обзор компонентов Borland Delphi](#_Toc201560027)

11

Лист

1.3.1 Описание компонентаTEdit

TEdit- стандартный управляющий элемент Windows для ввода. Он может быть использован для отображения короткого фрагмента текста и позволяет пользователю вводить текст во время выполнения программы. Значок компонента Edit находится на вкладке Standart в соответствии с рисунком 6.

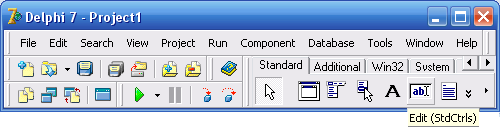


Рисунок 6 – Командная кнопка – компонент Edit

**1.3.1.1 Свойства объекта TEdit**

В таблице 2 перечислены основные свойства компонента Edit — поля ввода-редактирования.

Таблица 2 – Свойства компонента TEdit

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| Name | Имя компонента. Используется в программе для доступа к компоненту и его свойствам, в частности — для доступа к тексту, введенному в поле редактирования |
| Text | Текст, находящийся в поле ввода и редактирования |
| Left | Расстояние от левой границы компонента до левой границы формы |
| Top | Расстояние от верхней границы компонента до верхней границы формы |
| Height | Высота поля |
| Width | Ширина поля |
| Font | Шрифт, используемый для отображения вводимого текста |
| ParentFont | Признак наследования компонентом характеристик шрифта формы, на которой находится компонент. Если значение свойства равно True, то при изменении свойства Font формы автоматически меняется значение свойства Font компонента |

1.3.2 Описание компонентаTLabel

TLabelслужит для отображения текста на экране. Вы можете изменить шрифт и цвет метки, если дважды щелкнете на свойство Font в Инспекторе Объектов. Вы увидите, что это легко сделать и во время выполнения программы, написав всего одну строчку кода. Значок компонента Label находится на вкладке Standart в соответствии с рисунком 7.

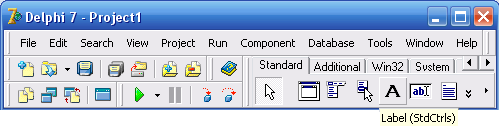


Рисунок 7 – Командная кнопка – компонент Label

**1.3.2.1 Свойства объекта TLabel**

Свойства компонента TLabel перечислены в таблице 3.

Таблица 3 – Свойства компонента TLabel

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| Name | Имя компонента. Используется в программе для доступа к компоненту и его свойствам |
| Caption | Отображаемый текст |
| Font | Шрифт, используемый для отображения текста |
| ParentFont | Признак наследования компонентом характеристик шрифта формы, на которой находится компонент. Если значение свойства равно True, текст выводится шрифтом, установленным для формы |
| AutoSize | Признак того, что размер поля определяется его содержимым |
| Left | Расстояние от левой границы поля вывода до левой границы формы |
| Top | Расстояние от верхней границы поля вывода до верхней границы формы |
| Height | Высота поля вывода |
| Width | Ширина поля вывода |

Лист

12

Следует обратить внимание на свойства Autosize и Wordwrap. Эти свойства нужно использовать, если поле вывода должно содержать несколько строк текста. После добавления к форме компонента Label значение свойства Autosize равно True, т. е. размер поля определяется автоматически в процессе изменения значения свойства Caption.

Если вы хотите, чтобы находящийся в поле вывода текст занимал несколько строк, то надо сразу после добавления к форме компонента Label присвоить свойству Autosize значение False, свойству Wordwrap — значение True. Затем изменением значений свойств Width и Height нужно задать требуемый размер поля. Только после этого можно ввести в свойство Caption текст, который должен быть выведен в поле.

**1.3.3 Описание компонента TButton**

TButtonпозволяет выполнить какие-либо действия при нажатии кнопки во время выполнения программы. В Delphi все делается очень просто. Поместив TButton на форму, Вы по двойному щелчку можете создать заготовку обработчика события нажатия кнопки. Далее нужно заполнить заготовку кодом:

Procedure TForm1.Button1Click (Sender: TObject);

Begin

…

end;

Значок компонента Button находится на вкладке Standart в соответствии с рисунком 8.

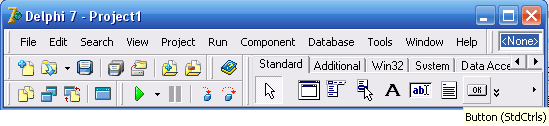


Рисунок 8 – Командная кнопка – компонент Button

**1.3.3.1 Свойства объекта TButton**

В таблице 4 перечислены некоторые свойства компонента TButton.

13

Лист

Таблица 4 – Свойства компонента TButton

|  |  |
| --- | --- |
| **Свойство** | **Описание** |
| Caption | Текст на кнопке |
| Enabled | Признак доступности кнопки. Кнопка доступна, если значение свойства равно True, и недоступна, если значение свойства равно False |
| Left | Расстояние от левой границы кнопки до левой границы формы |
| Top | Расстояние от верхней границы кнопки до верхней границы формы |
| Height | Высота кнопки |
| Width | Ширина кнопки |
| Name | Имя компонента. Используется в программе для доступа к компоненту и его свойствам |

Все описанные выше компоненты Delphi в данной курсовой работе использовались для создания программы.

**Программирование программы**

**2.1 Программа**

Программа, работающая на компьютере, нередко отождествляется с самим компьютером, т.к. человек, использующий программу, «вводит в компьютер» исходные данные, как правило, при помощи клавиатуры, а компьютер «выдаёт результат» на экран, на принтер или в файл. На самом деле, преобразование исходных данных в результате выполняет процессор компьютера. Процессор преобразует исходные данные в результате по определённому алгоритму, который, будучи записан на специальном языке, называется программой. Таким образом, чтобы компьютер выполнил некоторую работу, необходимо разработать последовательность команд, обеспечивающую выполнение этой работы, или, как говорят, написать программу.

**2.2 Постановка задачи**

Написать программу, которая проверяет, является ли введенная с клавиатуры квадратная матрица «магическим» квадратом.

Лист

14

**2.3 Этапы разработки программы**

Выражение «написать программу» отражает только один из этапов создания компьютерной программы, когда разработчик программы (программист) действительно пишет команды (инструкции) на бумаге или при помощи текстового редактора.

Программирование – это процесс создания (разработки) программы, который может быть представлен последовательностью следующих шагов:

1. Спецификация (определение, формулирование требований к программе).

2. Разработка алгоритма.

3. Кодирование (запись алгоритма на языке программирования).

4. Отладка.

5. Тестирование.

6. Создание справочной системы.

7. Создание установочного диска (CD-ROM).

**2.3.1 Спецификация**

Спецификация, определение требований к программе — один из важнейших этапов, на котором подробно описывается исходная информация, формулируются требования к результату, поведение программы в особых случаях (например, при вводе неверных данных), разрабатываются диалоговые окна, обеспечивающие взаимодействие пользователя и программы.

На этом этапе я подобрал информацию необходимую для создания программы. Определил, как должен выглядеть результат (Рисунок 9).

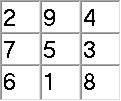


Рисунок 9 – «магический» квадрат

15

Лист

**2.3.2 Разработка алгоритма**

На этапе разработки алгоритма необходимо определить последовательность действий, которые надо выполнить для получения результата. Если задача может быть решена несколькими способами и, следовательно, возможны различные варианты алгоритма решения, то программист, используя некоторый критерий, например скорость решения алгоритма, выбирает наиболее подходящее решение. Результатом этапа разработки алгоритма является подробное словесное описание алгоритма или его блок-схема.

На этом этапе я создал блок – схему будущей программы (Рисунок 9).

Начало

Заполняем матрицу

Да

Цикл

i=1 to 3

Нет

sum <> a[i,1]+a[i,2]+a[i,3]

Да

flag:=0

Возврат цикла

Нет перехода

Цикл

Да

i=1 to 3

Лист

16

Нет

sum <> a[1,i]+a[2,i]+a[3,i]

Да

flag:=0

Возврат цикла

Нет перехода

sum <> a[1,1]+a[2,2]+a[3,3]

Нет

Да

flag:=0

Нет

flag:=1

Да

Матрица является магическим квадратом

Матрица не является магическим квадратом

Конец

Рисунок 9 – блок - схема программы

Лист

17

**2.3.3 Кодирование**

После того как определены требования к программе и составлен алгоритм решения, алгоритм записывается на выбранном языке программирования. В результате получается исходная программа. На этом этапе я выбрал язык программирования Delphi и записал алгоритм:

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls;

type

TForm1 = class(TForm)

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Edit5: TEdit;

Edit6: TEdit;

Edit7: TEdit;

Edit8: TEdit;

Edit9: TEdit;

Label1: TLabel;

Button1: TButton;

Label3: TLabel;

Button2: TButton;

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

// Процедура нажатия на кнопку "Магический квадрат?"

procedure TForm1.Button1Click(Sender: TObject);

var

Лист

18

//матрица, которую проверяем на "магичность"

a:array[1..3,1..3]of integer;

flag, // флаг, магический квадрат или нет (=0 - нет, =1 - да)

i, // счётчик циклов

sum:integer; // сумма элементов строки матрицы

begin

// Заполняем матрицу

a[1,1]:=StrToInt(Edit1.Text);

a[1,2]:=StrToInt(Edit2.Text);

a[1,3]:=StrToInt(Edit3.Text);

a[2,1]:=StrToInt(Edit4.Text);

a[2,2]:=StrToInt(Edit5.Text);

a[2,3]:=StrToInt(Edit6.Text);

a[3,1]:=StrToInt(Edit7.Text);

a[3,2]:=StrToInt(Edit8.Text);

a[3,3]:=StrToInt(Edit9.Text);

//Проверяем заполненную матрицу на «магичность»

flag:=1; // предполагаем, что матрица магический квадрат

sum:=0; // пока сумма элементов строки матрицы = 0

//подсчитываем сумму

for i:=1 to 3 do

sum:=sum+a[1,i];

//проверяем все строки

for i:=1 to 3 do

if(sum <> a[i,1]+a[i,2]+a[i,3]) then

flag:=0;

//проверяем все столбцы

for i:=1 to 3 do

if(sum <> a[1,i]+a[2,i]+a[3,i]) then

flag:=0;

//проверяем диагонали

if(sum <> a[1,1]+a[2,2]+a[3,3]) then

flag:=0;

// ВЫВОДИМ РЕЗУЛЬТАТ

if (flag=1) then

label3.caption:='Матрица является Магическим квадратом'

else

label3.caption:='Матрица не является Магическим квадратом';

Лист

19

20

Лист

end;

// Процедура нажатия на кнопку закрыть

procedure TForm1.Button2Click(Sender: TObject);

begin

close //Закрыть

end;

end.

**2.3.4 Массивы**

При написании программы использовался массив. *Массив —* это структура данных, представляющая собой набор переменных одинакового типа, имеющих общее имя. Массивы удобно использовать для хранения однородной по своей природе информации, например, таблиц и списков.

**Объявление массива**

Массив, как и любая переменная программы, перед использованием должен быть объявлен в разделе объявления переменных. В общем виде инструкция объявления массива выглядит следующим образом:

*Имя:* **array** *[нижний\_индекс. .верхний\_индекс]* ***of*** *тип*

где:

***имя*** *—* имя массива;

***array*** — зарезервированное слово языка Delphi, обозначающее, что объявляемое имя является именем массива;

***нижний\_индекс***и ***верхний\_индекс*** — целые константы, определяющие диапазон изменения индекса элементов массива и, неявно, количество элементов (размер) массива;

***тип*** *—* тип элементов массива.

В нашем случай объявление массива выглядит так:

a:array[1..3,1..3]of integer, где integer-тип целых чисел.

Для того чтобы в программе использовать элемент массива, надо указать имя массива и номер элемента (индекс), заключив индекс в квадратные скобки. В качестве индекса можно использовать константу или выражение целого типа, например:

sum:=sum+a[1,i];

21

Лист

**Операции с массивами**

Типичными операциями при работе с массивами являются:

* вывод массива;
* ввод массива;
* поиск максимального или минимального элемента массива;
* поиск заданного элемента массива;
* сортировка массива.

**Вывод массива**

Под выводом массива понимается вывод на экран монитора (в диалоговое окно) значений элементов массива.

Если в программе необходимо вывести значения всех элементов массива, то для этого удобно использовать инструкцию for, при этом переменная-счетчик инструкции for может быть использована в качестве индекса элемента массива. Например:

for i:=1 to 3 do

**Ввод массива**

Под вводом массива понимается процесс получения от пользователя (или из файла) во время работы программы значений элементов массива.

"Лобовое" решение задачи ввода элементов массива — для каждого элемента массива создать поле ввода. Однако если требуется ввести достаточно большой массив, то такое решение неприемлемо. Представьте форму, например, с десятью полями редактирования!

Очевидно, что последовательность чисел удобно вводить в строку таблицы, где каждое число находится в отдельной ячейке. Ниже рассматриваются два варианта организации ввода массива с использованием компонентов StringGrid и Memo.

Но этими и остальными операциями я не воспользовался, так как не использовал их в программе.

**2.4 Анализ и тестирование**

Программа запускается с помощью файла Mag kv. После запуска открывается окно, в котором необходимо заполнить матрицу или покинуть программу (рисунок 10).

22

Лист

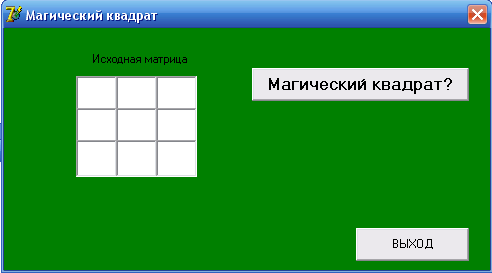


Рисунок 10 - Программа до щелчка мыши

После заполнения матрицы нажимаем на кнопку «Магический квадрат», и на экране выводится сообщение, является эта матрица «магическим» квадратом или нет (рисунок 11).

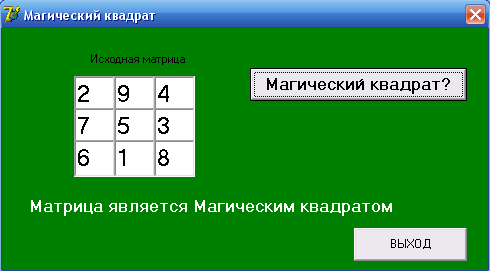


Рисунок 11 - Программа после щелчка мыши

На этом этапе программа была неоднократно протестирована, ошибок не выявлено.

**2.5 Создание справочной системы**

Из - за невысокой сложности программы, данный пункт не использовался.

23

Лист

**2.6 Создание установочного диска**

Из - за невысокой сложности программы, создание установочного диска нецелесообразно.

**Заключение**

24

Лист

В настоящее время в нашу жизнь широко внедряются информационные технологии. На сегодняшний день, наверное, не осталось такой области, где бы они ни применялись.

Постоянно растет объем и сложность обрабатываемой информации, требуются все новые и новые виды ее представления. Как показывает практика, большинство пользователей уже не представляет себе, как бы они выполняли свою работу, с которой прекрасно справлялись еще 5 – 10 лет назад, без помощи компьютера.

В ходе курсового проектирования разработана тестирующая программа. Проектирование выполнено в системе Delphi. В пояснительной записке:

* представлена информация о создании языка программирования;
* приведен теоретический материал о компонентах, используемых в программе;
* представлена блок-схема алгоритма работы программы;
* приведен анализ и тестирование программы с подробным описанием;

**Список литературы**

1. Чеснокова О.В., Delphi 2007. Алгоритмы и программы. Самоучитель. – М.: НТ Пресс, 2008 – 368 с.
2. Алексеев Е.Р., Чеснокова О.В., Турбо Паскаль 7.0, - М.: НТ Пресс, 2006 – 320 с.
3. Бобровский С., Delphi 5: учебный курс: Питер, 2001 – 640 с.
4. [Карчевская М.П.](mk:@MSITStore:D:\Разное\Угату\Программирование%20и%20основы%20алгоритмизации\учебники\Delphi!.chm::/Delphi/karch.htm), [Рамбургер О.Л.](mk:@MSITStore:D:\Разное\Угату\Программирование%20и%20основы%20алгоритмизации\учебники\Delphi!.chm::/Delphi/ramburger.htm), [Тархов С.В.](mk:@MSITStore:D:\Разное\Угату\Программирование%20и%20основы%20алгоритмизации\учебники\Delphi!.chm::/Delphi/tarkhov.htm), [Хамзина Е.А.](mk:@MSITStore:D:\Разное\Угату\Программирование%20и%20основы%20алгоритмизации\учебники\Delphi!.chm::/Delphi/hamzina.htm), Разработка приложений в среде Borland Delphi: лабораторный практикум по дисциплине "Информатика",- Уфа: УГАТУ, 2005.
5. Интернет-издание, 32 урока: Borland Delphi или о том, что лучше один раз увидеть, чем сто раз услышать.
6. Электронная книга. Архангельский А.Я., 100 компонентов общего назначения библиотеки Delphi 5.

Лист

25